STM JOURNALS

Journal of
VLSI Design Tools & Technology

ISSN: 2249-474X (Online)
ISSN: 2321-6492 (Print)
Volume 16, Issue 2, 2026
May–August
DOI (Journal): 10.37591/JoVDTT

https://journals.stmjournals.com/jovdtt

Review JoVDTT

Architectural Evolution of RISC Processors: From Early Research Prototypes to Modern Industrial Implementations

Aditya Chauhan[1], Mitesh Limachia[2,*], Narendra Chauhan[3], Purvang Dalal[4]

Abstract

This paper surveys the architectural evolution of Reduced Instruction Set Computer (RISC) architectures from early research prototypes to modern industrial processors, tracing four decades of design refinement across academic, embedded, server, mobile, and open-hardware ecosystems. Core ISA principles, pipeline evolution, compiler–hardware co-design, and quantitative performance modeling are analyzed through four representative industrial implementations: MIPS 74K, SPARC T4, ARM Cortex-A72, and SiFive U74 (RISC-V). Each processor is examined in terms of its unique microarchitectural features, target application domain, and position within the broader instruction-count, cycle-per-instruction, and clock-period performance framework. While execution engines have evolved substantially from simple in-order pipelines toward superscalar, out-of-order, and multithreaded designs fundamental RISC characteristics such as load/store execution discipline, ISA regularity, and register-centric computation remain persistent across generations. The survey further examines emerging challenges facing high-performance RISC implementations, including speculative-execution security vulnerabilities such as Spectre and Meltdown, the growing importance of energy-proportional computing under Dark Silicon constraints, and the rise of Domain-Specific Architectures (DSAs) enabled by the open and extensible RISC-V ecosystem. Comparative tables consolidate ISA-level and microarchitectural distinctions across six representative designs, offering a structured reference for researchers and practitioners. By bridging classical RISC tenets with modern execution, security, and efficiency constraints, this work provides a comprehensive technical synthesis of how different RISC families optimized their architectures for distinct computing domains while preserving the core principles that originally defined the RISC paradigm.

***Author for Correspondence**
Mitesh Limachia
E-mail: mitesh.ec@ddu,ac.in

[1]Student, Department of Electronics and Communication Engineering, Dharmsinh Desai University, Nadiad, Gujarat, India
[2]Associate Professor, Department of Electronics and Communication Engineering, Dharmsinh Desai University, Nadiad, Gujarat, India
[3]Assistant Professor, Department of Electronics and Communication Engineering, Dharmsinh Desai University, Nadiad, Gujarat, India
[4]Professor and Head of Department, Department of Electronics and Communication Engineering, Dharmsinh Desai University, Nadiad, Gujarat, India

Received Date: July 31, 2026
Accepted Date: August 12, 2026
Published Date: August 20, 2026

Citation: Aditya Chauhan, Mitesh Limachia, Narendra Chauhan, Purvang Dalal. Architectural Evolution of RISC Processors: From Early Research Prototypes to Modern Industrial Implementations. Journal of VLSI Design Tools & Technology. 2026; 16(2): 29–39p.

Keywords: RISC, MIPS, SPARC, ARM, RISC-V, pipeline architecture, CPI, load/store architecture

INTRODUCTION

The Reduced Instruction Set Computer (RISC) paradigm has become the dominant architectural philosophy in modern microprocessors. RISC principles, such as fixed-length instructions, load/store execution, and extensive register usage, enable simplified datapaths and deep pipelining, allowing many instructions to be executed in a single cycle [1].

By the mid-2010s, RISC processors powered nearly all mobile devices and most shipped computing systems [2]. As Moore's law scaling

slowed, architectural innovation increasingly shifted toward microarchitecture efficiency, parallelism, and specialization. The emergence of open Instruction Set Architectures (ISAs), particularly RISC-V, has further revitalized processor research by enabling collaborative innovation without proprietary constraints.

While numerous surveys have documented the microprocessor history [2], few provide a cross-generational technical synthesis of how core RISC principles have fractured and specialized to meet contemporary demands. Patterson's retrospective [2] provides an excellent historical context, and Waterman [3] details the RISC-V rationale. However, top-tier venues require comparative analysis across active ISAs (ARM, MIPS, SPARC, RISC-V) concerning their microarchitectural divergence, security implications, and energy economics.

This survey paper demonstrates comprehensive evolution of RISC architectures by focusing on architectural principles rather than pure historical narration. Representative processors are analyzed to demonstrate how RISC concepts are adapted to different computing domains. Furthermore, we explore modern microarchitectural complexities, power-performance dynamics, security vulnerabilities such as speculative execution side-channels, and the rise of domain-specific architectures (DSAs) driven by RISC-V.

Related Work and Motivation

Foundations of RISC

The RISC paradigm emerged in the mid-1970s to simplify instruction sets for higher performance. IBM's 801 project demonstrated key RISC principles including load/store architecture, fixed-length instructions, and large register files, enabling simplified pipelining and faster clock rates [4].

Berkeley's RISC-I and RISC-II formalized these concepts using aggressive compiler scheduling and a 32-bit fixed instruction format. A major innovation was the introduction of windows, which reduced procedure-call memory traffic by overlapping register sets. Hazard management relies heavily on compiler scheduling and branch delay slots, shifting the complexity from hardware to software [5].

Collectively, these early systems established the architectural template still followed today: uniform instruction encoding, register-based computation, pipeline-friendly design, and compiler hardware co-optimization.

Industrial RISC Architectures

MIPS and MIPS 74K

MIPS extended the Berkeley RISC ideas into commercial high-performance systems using a classic five-stage pipeline (IF–ID–EX–MEM–WB). The architecture intentionally eliminated hardware interlocks, relying on compiler scheduling to resolve hazards, thereby simplifying hardware control logic [6, 7].

Unique Architectural Features

- Software-managed pipeline hazards
- Branch delay slot exposure
- Direct mapping between ISA and micro-operations

The MIPS 74 K core represents a modern evolution, introducing dual-issue superscalar execution, DSP extensions, and dynamic branch prediction for multimedia workloads [8].

Despite its increased complexity, the ISA retains a strict load/store discipline. Figure 1 illustrates the MIPS 74 K pipeline architecture, and Figure 2 demonstrates its core programmer model.

Applications and Relevance

MIPS 74 K processors are widely used in networking equipment, digital media devices, and embedded signal-processing systems, where deterministic execution and efficient DSP capabilities are essential.

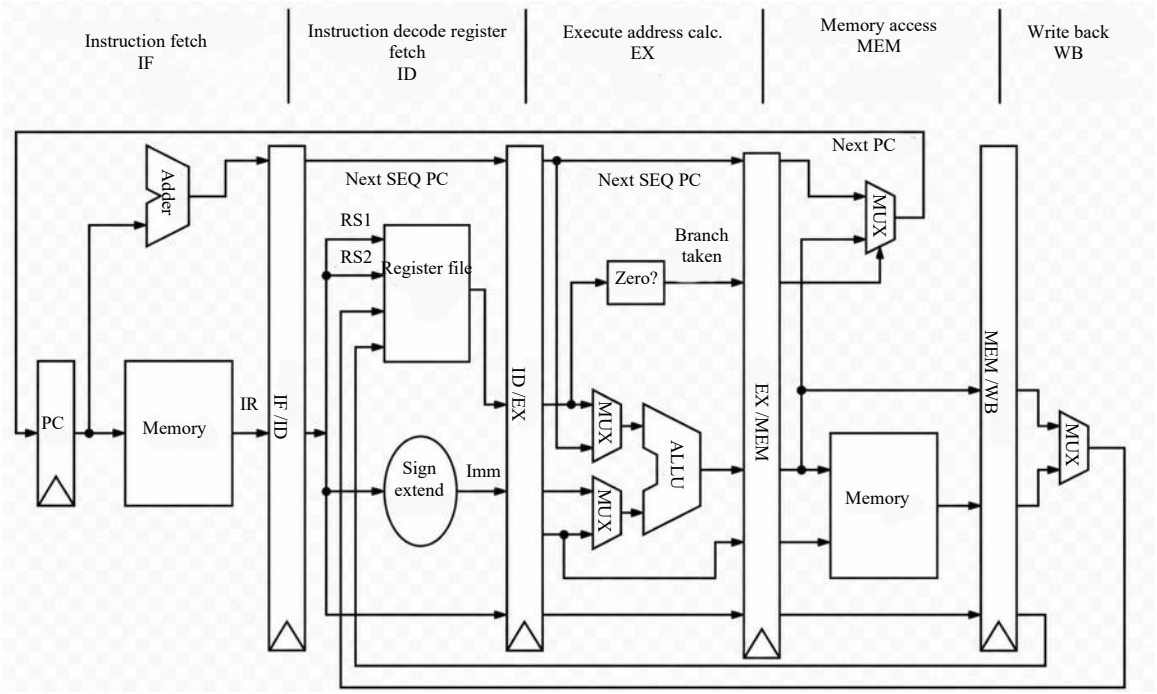

Figure 1. MIPS 74K pipeline organization.

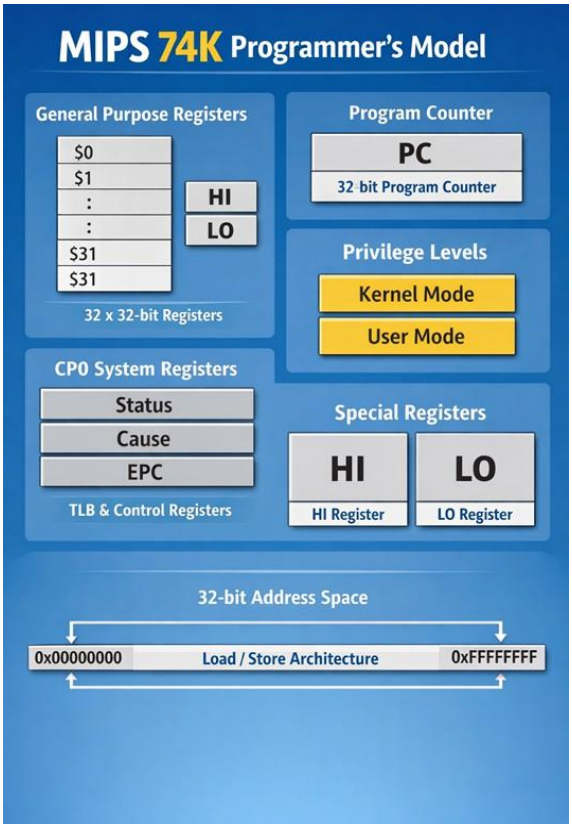

Figure 2. MIPS 74K programmer model.

SPARC and SPARC T4

SPARC adopted a load/store ISA with hardware-supported register windows, reducing the procedure-call overhead while maintaining compiler transparency [9].

Unique Architectural Features

- Register window mechanism
- Hardware-managed context switching
- Integrated cryptographic accelerators
- High thread-level parallelism

The Oracle SPARC T4 marked a major transition from throughput-oriented in-order cores to out-of-order (OoO) execution with deep pipelines and advanced branch prediction [10]. The processor integrates eight cores supporting up to 64 hardware threads context-switched in hardware with built-in encryption acceleration. Figure 3 details the massive multithreading in the SPARC T4 pipeline, and Figure 4 highlights the complex register window scheme.

Applications and Relevance

SPARC T4 targets enterprise servers, virtualization platforms, and mission-critical datacenter workloads that require reliability, scalability, and secure processing.

ARM and Cortex-A72

ARM architecture prioritizes energy efficiency and system integration. While the classic 32-bit ARM architecture relies heavily on conditional execution to reduce branch penalties, modern 64-bit implementations (ARMv8-A), such as Cortex-A72, have largely deprecated full predication in favor of conditional select instructions to simplify out-of-order pipeline decoding.

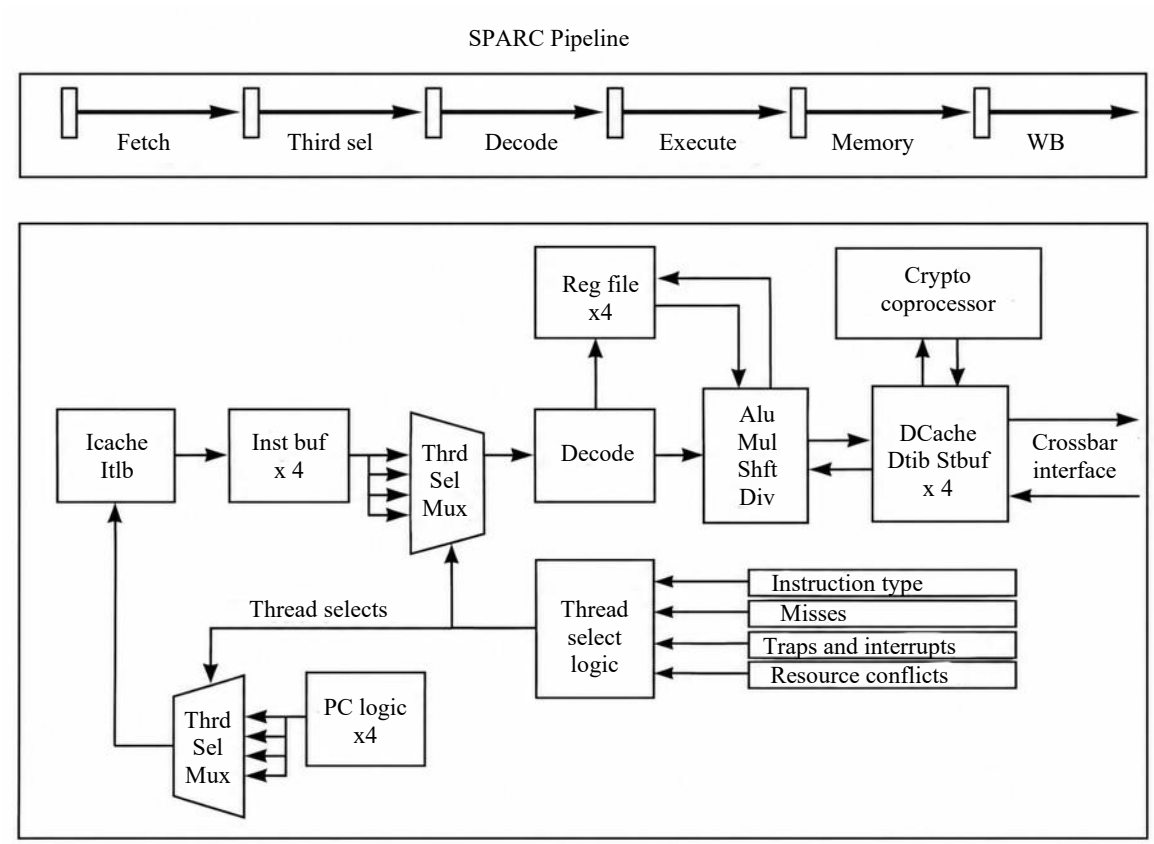

Figure 3. SPARC T4 Pipeline.

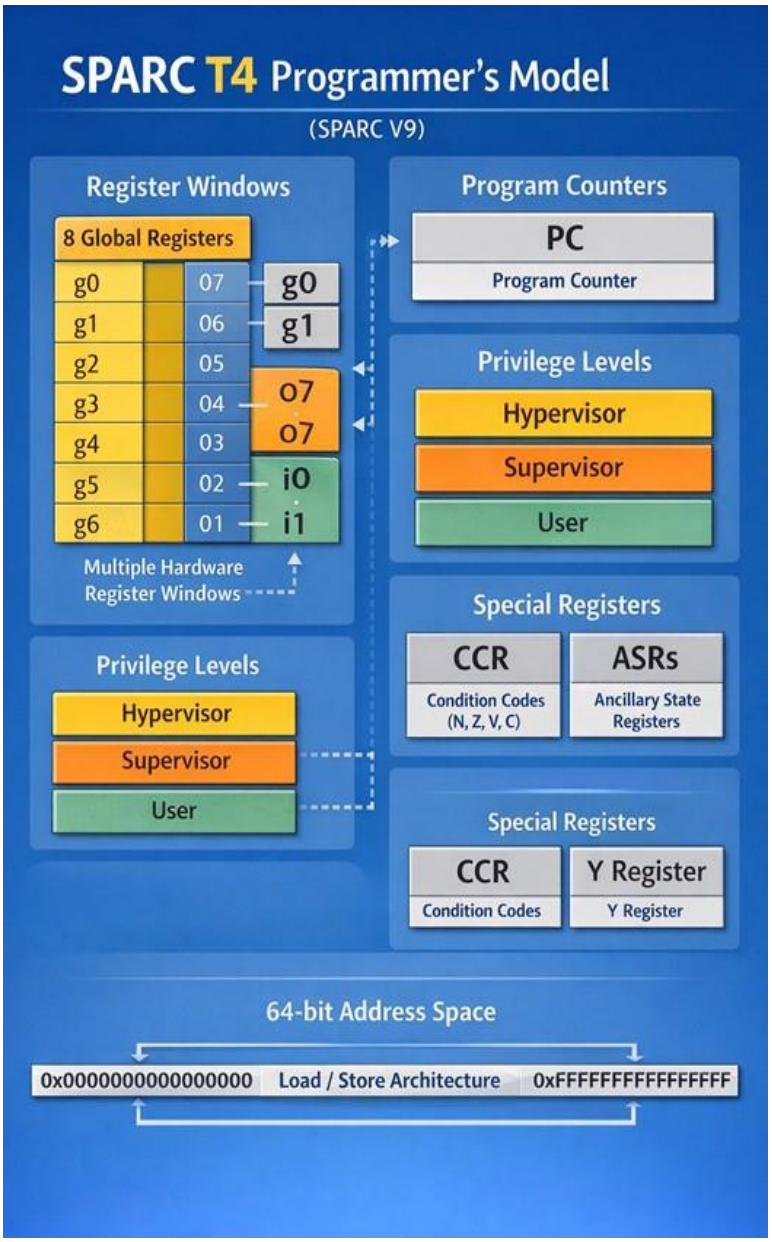

Figure 4. SPARC T4 programmer model.

Unique Architectural Features

- Conditional instruction execution
- Power-efficient pipeline design
- Heterogeneous multicore scalability

The Cortex-A72 represents a high-performance ARMv8-A implementation featuring superscalar out-of-order execution optimized for mobile and edge computing workloads. Its architecture heavily relies on advanced dynamic branch prediction and highly associative caches to maintain pipeline utilization in energy-constrained environments. Figure 5 shows the extensive OoO superscalar pipeline, and Figure 6 shows the AArch64 programmer model.

Applications and Relevance

Cortex-A72 processors power smartphones, embedded AI platforms, and single-board computers, balancing their performance with strict power constraints.

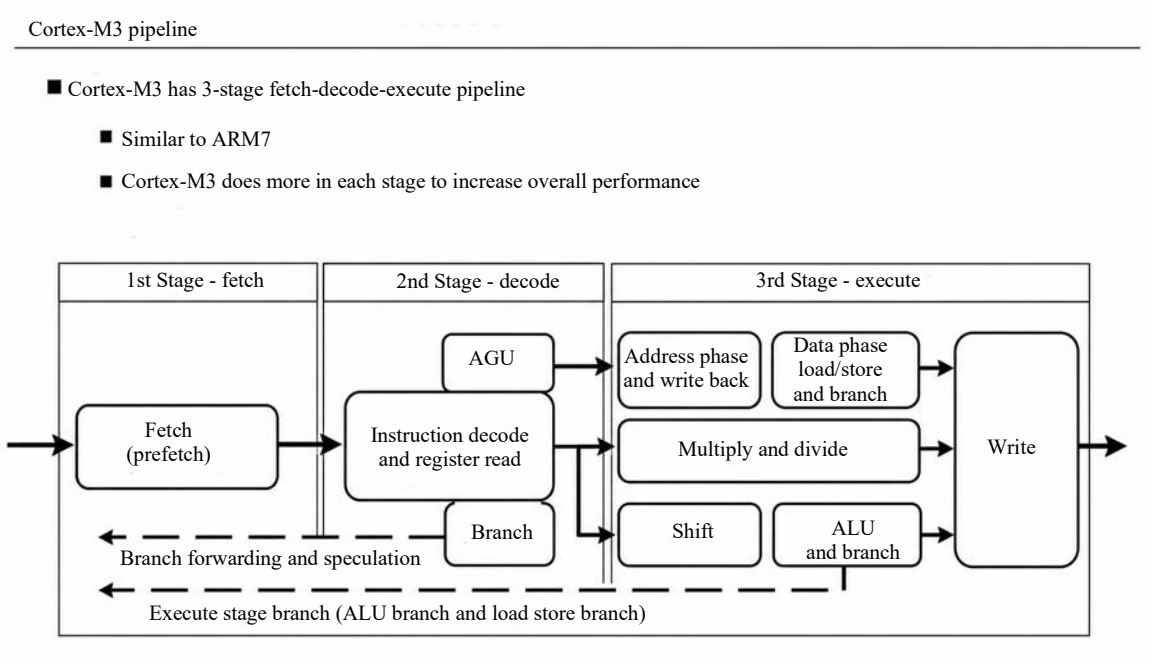

Figure 5. ARM Cortex-A72 pipeline.

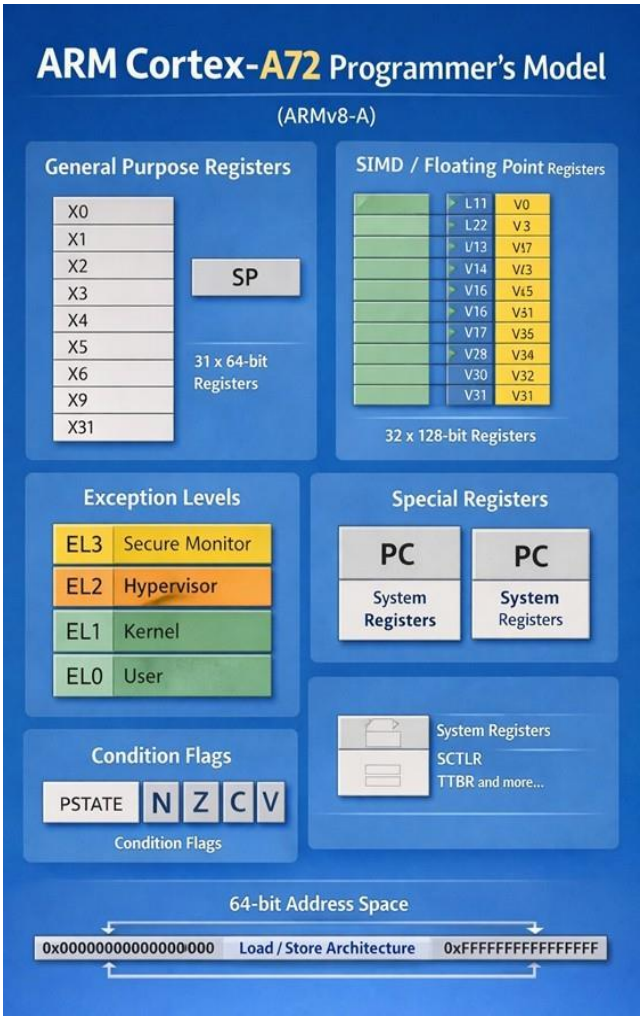

Figure 6. ARM cortex-A72 programmer model.

RISC-V and SiFive U74

RISC-V represents a minimalist and modular continuation of RISC philosophy. The base ISA provides only essential integer operations, whereas the extensions add floating-point, atomic, and vector capabilities [8].

Unique Architectural Features

- Open and royalty-free ISA
- Modular extension framework
- Custom instruction extensibility

The SiFive U74 core implements RV64GC with a modern dual-issue in-order pipeline optimized for Linux-capable embedded systems while preserving architectural simplicity. Figure 7 details the streamlined U74 pipeline, and Figure 8 shows a clean base integer programmer model.

Applications and Relevance

RISC-V processors increasingly appear in IoT devices, research platforms, accelerators, and emerging commercial SoCs due to ecosystem openness and customization flexibility.

COMPARISON OF RISC ARCHITECTURES

Architectural evolution in RISC processors can be analyzed using the classical performance model:

$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock Period} \quad (1)$$

This relationship highlights how RISC architectures improved performance not by increasing instruction complexity, but by optimizing execution efficiency across all three parameters [1].

Instruction Count (IC)

Early RISC systems increased the instruction count relative to CISC designs because of simplified instructions. However, uniform instruction encoding enables aggressive compiler optimization and improves pipeline utilization. Modern processors such as Cortex-A72 and SiFive U74 compensate through instruction-level parallelism and improved branch prediction, reducing effective execution overhead.

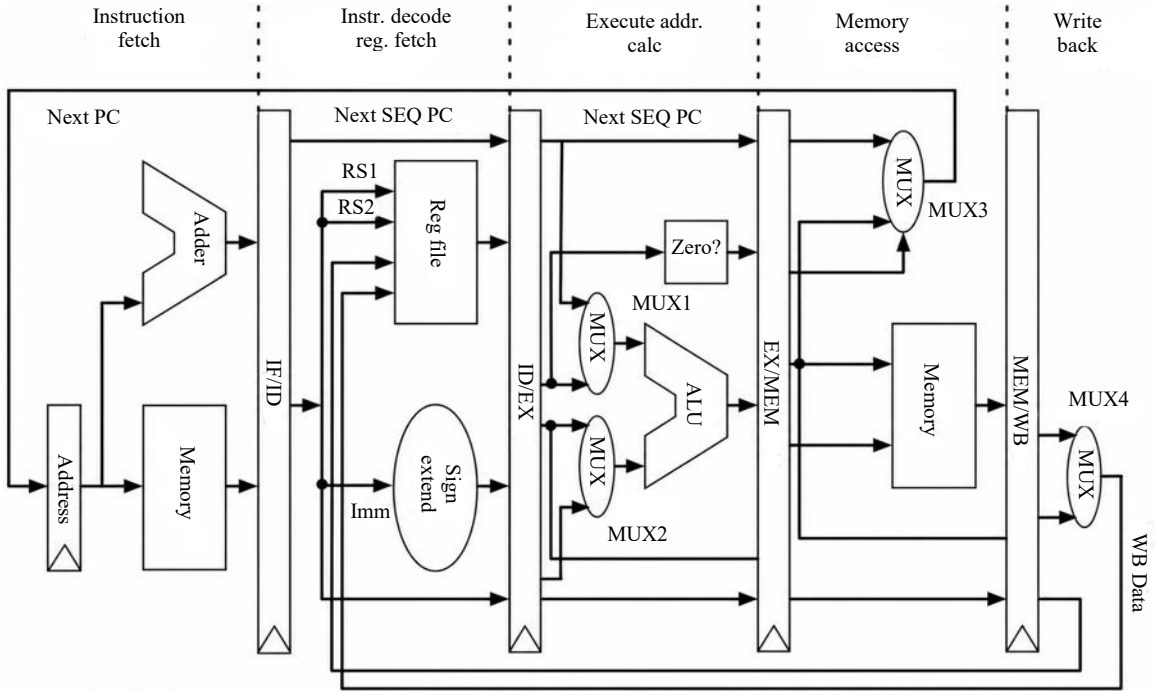

Figure 7. SiFive U74 (RISC-V) pipeline.

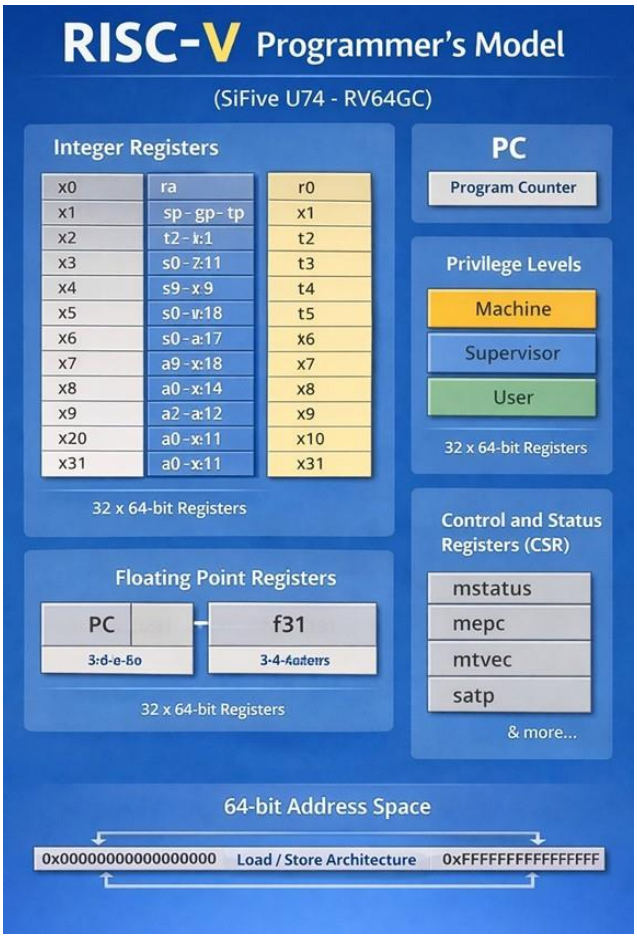

Figure 8. RISC-V U74 programmer model.

Cycles Per Instruction (CPI)

A major goal of RISC design is to achieve CPI ≈ 1 under ideal conditions. Early processors, such as MIPS, achieved this through compiler-scheduled pipelines and branch delay slots. Later, the architecture shifted toward dynamic hazard resolution.

- MIPS 74K reduces CPI using superscalar dual-issue execution.
- SPARC T4 improves throughput via multithreading and out-of-order execution.
- Cortex-A72 minimizes stalls through speculative execution.
- SiFive U74 balances CPI and power using efficient in-order pipelines.

Thus, CPI reduction evolved from static scheduling to hardware-driven parallelism.

Clock Period and Pipeline Depth

Simplified datapaths allow higher clock frequencies in early RISC processors. As transistor scaling slowed, performance gains shifted toward deeper pipelines and parallel execution units rather than raw frequency increases.

Server-oriented processors, such as SPARC T4, emphasize throughput, whereas ARM Cortex-A72 prioritizes performance-per-watt. RISC-V designs offer flexible trade-offs depending on the implementation goals.

Energy and Performance Trade-offs

Modern RISC evolution demonstrates a shift from the maximum frequency toward energy-efficient computation. ARM architectures dominate mobile systems because of their superior performance-per-

watt, whereas SPARC targets reliability and throughput. RISC-V enables domain-specific acceleration through ISA extensions, allowing for the optimization of specialized workloads.

Overall, RISC performance evolution reflects balanced optimization across IC, CPI, and clock period rather than reliance on a single scaling factor. Energy efficiency, driven by Dark Silicon limits, now dictates microarchitectural choices more than absolute frequency scaling.

To highlight architectural evolution, Tables 1 and 2 compare representative RISC designs across ISA structure, pipeline organization, compiler interaction, and application relevance. This comparison illustrates how core RISC principles remain consistent, whereas implementation strategies diverge based on target computing domains.

Modern Challenges and Future Trends

Security Implications: Speculative Execution

The relentless pursuit of CPI ≈ 1 through OoO execution and deep branch prediction introduces severe security vulnerabilities. The discovery of Spectre and Meltdown exposed how side-channel attacks could exploit speculative execution within high-performance RISC processors (e.g., ARM Cortex-A series, SPARC). Modern RISC architectures are now forced to implement microarchitectural mitigations that negatively impact CPI, illustrating a new fundamental trade-off between performance and security isolation. The open nature of RISC-V enables novel, hardware-enforced secure enclaves (e.g., Keystone) to address these threats at the ISA level.

Table 1. ISA and architectural features of selected RISC designs.

Feature	IBM 801	RISC-I/II	MIPS	SPARC V8	ARM	RISC-V
Load/Store Architecture	Yes	Yes	Yes	Yes	Yes	Yes
Fixed-Length Instructions	Yes	32-bit	32-bit	32-bit	32-bit	32-bit (base)
Register File Size	16–32	32	32	32	16–32	32
Register Windows	No	Yes	No	Yes	No	No
Conditional Execution	No	Limited	No	No	Yes	Ext.-based
Branch Delay Slot	No	Yes	Yes	Yes	No	No
Pipeline Interlocks	Hardware	Hardware	Software	Hardware	Hardware	Hardware
Superscalar/OoO Support	No	No	Later	Later	Yes	Emerging
Compiler Dependence	Moderate	High	High	Moderate	Moderate	Hybrid
Primary Ecosystem	Research	Academic	Embedded	Servers	Mobile	Open

Table 2. Microarchitectural and application relevance comparison.

Aspect	IBM 801	RISC-I/II	MIPS	SPARC	ARM	RISC-V
Pipeline depth (stages)	4–5	5	5	5	3–5	5+
Typical ideal CPI	~1	~1	~1	~1	1–2	~1
DMIPS/MHz (Approx.)	N/A	N/A	~1.8	N/A	4.0–5.7	2.5
Branch handling	Software	Delay slot	Delay slot	Delay slot	Prediction	Prediction
Data hazard resolution	HW Interlocks/Fwd.	HW Interlocks/Fwd.	Compiler	Hardware	Forwarding	Hardware
Execution style	In-order	In-order	Superscalar	OoO/MT	OoO	In-order/OoO
Power/area focus	Logic	Logic	Frequency	Throughput	Energy	Flexible
Representative core	IBM 801	RISC-II	MIPS 74K	SPARC T4	Cortex-A72	SiFive U74
Application domain	Research	Academic	Embedded	Servers	Mobile/Edge	IoT to HPC
Current relevance	Historical	Historical	Legacy	Niche	Dominant	Growing

Domain-Specific Architectures (DSAs)

As Moore's law slows, the general-purpose computing efficiency plateaus. The future of RISC lies in Domain-Specific Architectures (DSAs), which are strongly championed by the RISC-V ecosystem. By utilizing custom instruction extensions, designers can create highly specialized accelerators for AI/ML, cryptography, and graph processing while maintaining a standard software stack for the control logic.

CONCLUSION

RISC architecture demonstrates decades of co-evolution between ISA design and microarchitecture. The foundational ideas introduced by IBM 801 and Berkeley RISC remain central despite the dramatic advances in execution engines. Industrial architecture validated these principles across diverse domains, while RISC-V generalizes them into an open modular framework likely to influence future computing systems.

The comparative analysis presented in this survey shows that, despite four decades of implementation diversity, the defining characteristics of RISC – load/store execution discipline, uniform and regular instruction encoding, and register-centric computation–have proven remarkably durable. What has changed substantially is not the ISA philosophy itself but the microarchitectural machinery built around it: in-order pipelines have given way to superscalar and out-of-order execution, static compiler scheduling has been supplemented (and in many designs largely replaced) by dynamic hardware prediction, and single-threaded throughput has increasingly been traded for multithreading, speculative execution, and heterogeneous core composition. Each representative processor examined here, MIPS 74 K, SPARC T4, ARM Cortex-A72, and SiFive U74, illustrates a different point along this design continuum, shaped by the performance, power, and market constraints of its target domain rather than by any fundamental disagreement over RISC principles.

Simultaneously, this survey highlights that the RISC design space is no longer a single evolutionary line but a fractured ecosystem of specialized branches. Embedded and networking workloads continue to favor compact, deterministic pipelines; server and datacenter workloads prioritize thread-level parallelism, reliability, and hardware-assisted security; mobile and edge platforms optimize aggressively for performance-per-watt; and the open RISC-V ecosystem is increasingly defined by extensibility and domain-specific customization rather than a fixed reference implementation. The pipeline now makes it easy to specialize, extend, and verify at the ISA level.

The security and efficiency discussed in this study are also likely to shape the next phase of RISC evolution more than raw frequency or core-count scaling. As speculative execution vulnerabilities continue to surface in high-performance out-of-order designs, architects will need to balance CPI-oriented optimizations against verifiable isolation guarantees, an area where the open and inspectable nature of RISC-V provides a distinct research advantage over closed proprietary ISAs. Simultaneously, the slowing of traditional transistor scaling and the rise of Dark Silicon constraints make Domain-Specific Architectures rather than further general-purpose pipeline complexity the most promising avenue for continued performance growth, particularly for AI/ML, cryptographic, and graph-processing accelerators built on customizable RISC-V cores.

REFERENCES

1. Patterson DA, Ditzel DR. The case for the reduced instruction set computer. In: Readings in Computer Architecture. San Francisco (CA): Morgan Kaufmann; 2000. p.135–43.
2. Ganek AG, Corbi TA. The dawning of the autonomic computing era. IBM Syst J. 2003;42(1):5–18.
3. Tamir Y, Tremblay M. High-performance fault-tolerant VLSI systems using micro rollback. IEEE Trans Comput. 1990;39(4):548–54.
4. Hennessy J. VLSI processor architecture. IEEE Trans Comput. 1984;100(12):1221–46.

5. Hennessy J, Jouppi N, Baskett F, Gross T, Gill J. Hardware/software tradeoffs for increased performance. ACM SIGPLAN Not. 1982;17(4):2–11.
6. Garner RB. The scalable processor architecture (SPARC). In: The SPARC Technical Papers. New York (NY): Springer New York; 1991. p. 3–31.
7. Patterson D. Reduced instruction set computers then and now. Computer. 2017;50(12):10–2.
8. Waterman A, Lee Y, Patterson D, Asanovic K. The RISC-V instruction set manual. Volume I: User-level ISA. Version 2. 2014 May 6;2:1–79.
9. Novkovic T, Lukac Z, Jovanovic P, Kastelan I. Graphic library optimization for MIPS architecture. Elektron Elektrotech. 2020;26(2):69–76.
10. Shin JL, Golla R, Li H, Dash S, Choi Y, Smith A, et al. The next generation 64b SPARC core in a T4 SoC processor. IEEE J Solid-State Circuits. 2013;48(1):82–90.

